REMARKS

Claims 1, 6, 7, 9, 10, 15-17 and 22-32 are pending in the present application. Claims 1, 6, 7, 9, 10, 17 and 24-32 have been amended herewith. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 101

The Examiner rejected Claims 1, 6, 7, 9, 10, 15-17 and 22-32 under 35 U.S.C. § 101 as being directed towards non-statutory subject matter. This rejection is respectfully traversed.

With respect to Claims 1, 6, 7, 9 and 24-26, the Examiner states that such claims are directed to method steps which can be practiced mentally in conjunction with pen and paper, and suggests that the claims be amended to recite 'computer implemented method' instead of 'method'. Applicants have amended such claims accordingly, and thus the 35 U.S.C. § 101 of Claims 1, 6, 7, 9 and 24-26 has been overcome.

With respect to Claims 10, 15, 16 and 30-32, the Examiner states that such claims are not limited to tangible embodiments. Applicants have amended such claims herewith to expressly recite that such claims are limited to tangible embodiments, and thus the 35 U.S.C. § 101 of Claims 10, 15, 16 and 30-32 has been overcome.

With respect to Claims 17 and 27-29, the Examiner states that the system is, at best, a software system per se, failing to be tangibly embodied or include any recited hardware as part of the system. Applicants have amended such claims to expressly recite hardware as being a part of the claimed system, and thus the 35 U.S.C. § 101 of Claims 17 and 27-29 has been overcome.

II. 35 U.S.C. § 103, Obviousness

A. The Examiner rejected Claims 1, 10, 17, 24, 26, 27, 29, 30 and 32 under 35 U.S.C. § 103 as being unpatentable over Hussey (US Pat. Application Publication 2005/0086641). This rejection is respectfully traversed.

Claim 1 is directed to a method for correcting a path sequence of an environment variable in a data processing system, *the path sequence specifying an order for searching directories for locating executable code within the data processing system*. The method includes steps of (i) monitoring the data processing system for a change effecting the path sequence of the environment variable, where the *environment variable* is enabled and being used by the data

processing system to *specify the order for searching a plurality of different directories* within the data processing system, (ii) determining whether any duplicate files exist in any of the directories identified by the path sequence, and (iii) altering the path sequence of the environment variable to ensure that a proper executable file is found and executed when selected by one of a user and a running application program. As can be seen, the path sequence of an environment variable is altered, and this path sequence that is altered specifies an order for searching directories to locate executable code. While the teachings of the cited Hussey reference are directed to a technique for automatically updating a computer registry, the particulars of what is updated and how it is determined whether or not to update the registry are substantially different from Claim 1. Importantly, the cited Hussey reference teaches that an application module is originally booted, and then a determination is made – after the application module has been booted – as to whether any registry parameters that are used by the booted application program need to be modified (page 1, paragraphs [0017] and [0019]; Figure 3, block 310). In contrast, the present invention of Claim 1 is directed to altering a path sequence such that the proper application program itself can be located – the application program is not booted and running (per Hussey) but is trying to be found in order to be executed (per Claim 1). Applicants have amended Claim 1 to further emphasize this distinction. In addition, as Hussey requires that the application program itself *already be booted* in order to determine what registry entries used by such application program may need to be changed (as further articulated below with respect to Claim 24), there would have been no reason or other motivation to modify such teachings to alter an environment variable that specifies a search order for locating an executable program *prior to executing* the executable program.

Still further with respect to Claim 1, Applicants urge that the cited Hussey reference does not teach or suggest *altering of a path sequence that specifies an order for searching directories to locate executable code.* Rather, the cited reference teaches that a variable containing an specific reference *to the file name itself* – and not a *search order for searching a plurality of directories* to locate the file – is altered (page 4, paragraphs [0046] and [0047]). While the reference alludes to changing a path to a file such that the file can be located if moved from its original location, the path that is changed is *the explicit path to the file itself*, and includes the file name of the file (page 4, paragraph [0048]). In contrast, the present invention modifies a search order for *searching directories* in order to locate a file. Because the cited reference teaches that

the path is an explicit reference to the actual location of the file, as it includes both the directory and file name of where the file is, there would be no need to also modify any type of directory search order, and a plurality of directories do not have to be search to locate the file due to its explicit reference to the specific file name of where the file is specifically located. Thus, Claim 1 is further shown to not be obvious in view of the cited reference, as there are further claimed features not taught or suggested by the cited reference.

Still further with respect to Claim 1, such claim recites "responsive to determining that duplicate files do exist, altering the path sequence of the environment variable to ensure that a proper executable file is found and executed when selected by one of a user and a running application program". The Examiner acknowledges that the cited Hussey reference does not teach any type of duplicate file determination, but states that this claimed feature would have been obvious in view of Hussey since Hussey teaches that a determination is made as to whether a new operating system has been installed or an operating system has been reinstalled, and therefore "there would be duplicate files exist in the other directories which have been renamed by the user". Applicants urge error, in that even if such assertion were true, it does not establish the particular 'responsive action' that occurs based upon a determination that duplicate files exist. Specifically, the cited reference does not teach or otherwise suggest that a *path sequence* of an environment variable (which is specifically defined in the claim to be used for specifying an order for searching multiple directories in order to locate executable code) *is altered in response to a duplicate file determination*. Thus, the Examiner has failed to establish a prima facie showing of obviousness with respect to Claim 1[1], as the Examiner merely alleges that duplicate files *may exist*. In addition, the fact that a prior art device could be modified so as to produce the claimed device is not a basis for an obviousness rejection unless the prior art suggested the desirability of such a modification. *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984). As the cited reference does not teach or otherwise suggest any type of path sequence – and in fact has no need for a path sequence as executable files are explicitly specified as to there exact location (including both directory and file name, see page 4 paragraphs [0046 and 0047]) – there is no suggestion of any desirability to modify the teachings of Hussey to

[1] To establish prima facie obviousness of a claimed invention, *all of the claim limitations* must be taught or suggested by the prior art. MPEP 2143.03. *See also, In re Royka*, 490 F.2d 580 (C.C.P.A. 1974) (emphasis added by Applicants).

include such a path sequence or modification of such (missing) path sequence in response to a duplicate file determination. Thus, Claim 1 is further shown to be non-obvious in view of the cited reference.

There is yet another reason of why a person or ordinary skill in the art would not have been motivated to modify the teachings of Hussey in accordance with the claimed invention recited in Claim 1. Because Hussey loads or boots-up the application *prior to* determining if registry changes need to made to application-related variables maintained in a registry, the application that is booted-up has already been located, so there would have been no reason to modify a path sequence that specifies a directory search order to locate the executable file as it has already been located and booted. This further evidences non-obviousness of the present invention recited in Claim 1.

Thus, Claim 1 is shown to have been erroneously rejected for numerous reasons as identified above.

Applicants traverse the rejection of Claims 10 and 17 for similar reasons to those given above with respect to Claim 1.

With respect to Claim 24, such claim recites a method for managing environment variables in a data processing system, with an environment variable manager that is automatically invoked to correct a path sequence. The method includes steps of (i) automatically invoking an environment variable manager upon occurrence of at least one of occurring events: a) a directory is deleted, b) a product is uninstalled on the data processing system, and c) a given environment variable is manually modified by a user; (ii) determining, by the environment variable manager, if any occurring event a), b) or c) causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching a plurality of different directories for locating executable code within the data processing system; and (iii) automatically correcting the affected path sequence if it is determined that the occurring event causes the modification. As can be seen, a determination is made as to whether any of several listed events causes a modification to an affected path sequence of any presently active environment variable, the path sequence *specifying an order for searching directories for locating executable code within the data processing system.* As a part of such determination, an environment variable manager is *automatically invoked* upon occurrence of at least one of occurring events: a) a directory is deleted, b) a product is uninstalled on the data processing

system, and c) a given environment variable is manually modified by a user. Applicants urge that the cited reference does not teach or suggest (1) automatically invoking such manager upon occurrence of the listed events, (2) determining whether a particular event (as listed in the claim) caused a modification to a path sequence of an environment variable, where the path sequence that is modified specifies an order for searching directories to locate executable code; or (3) the correcting of such path sequence (the path sequence that is corrected is a path sequence that specifies an order for searching directories to locate executable code). As to the automatic invocation step, this is a pro-active step where problems are corrected upon occurrence of the problem being created. In contrast, the cited reference teaches a reactive mode of correcting problems that may have previously been introduced by certain user actions such as moving or renaming files. This is a reactive mode of correction, as the executable program is first booted, and then a determination is made as to whether registry keys used by such booted program need to be modified. In addressing this aspect of Claim 24, the Examiner states the system automatically updates the registry when the program module is booted so that the paths contained in the registry remain valid, and therefore it would have been obvious to relate the concept of environment variable manager to the step of automatically updating "since it appears that Hussey's invention would perform the functions equally well as the environment manager as claimed". Applicants urge that to the contrary, Hussey system is predicated upon *first* booting the executable program and *then* correcting registry problems that may have previously been introduced by a user. This post-boot sequence is dictated by the fact that (A) Hussey requires the program to be booted first such that the cache associated with such program (the registration cache) can be searched to determine if changes need to be made to registry keys (page 2, paragraph [0027]; page 4, paragraph [0059]), (B) Hussey requires the program to be booted first such that the .srg files (the files used to define resources used by the booted application) is accessible, as it is built into the resource fork of the application itself so that it is assured of being accessible without fear of inadvertent moving or deleting of such file (page 4, paragraph [0053]); and (C) Hussey uses the fact that files have out-of-date time/stamps when a program is booted to determine whether to modify a registry (page 4, paragraph [0054]; page 5, paragraph [0059]), and this out-of-date mechanism would not work for a pro-active system where problems are detected at the time they actually are introduced into the system. Quite simply, the teachings of the cited reference require the application program that needs its registry entries corrected to

first be booted prior to such registry correction, whereas the present invention makes corrections at the time the problem is actually introduced (through such actions as (a) a directory is deleted, (b) a product is uninstalled on the data processing system, and (c) a given environment variable is manually modified by a user. One system (Hussey) is re-active after executable program boot, whereas the other (Claim 24) is pro-active before executable program boot. Although a device may be capable of being modified to run the way [the patent applicant's] apparatus is claimed, there must be a suggestion or motivation *in the reference* to do so. *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990). There is simply no suggestion or motivation in the cite Mussey reference to modify such teachings in accordance with the claimed invention, as the entire detection mechanism for determining whether the registry needs to be modified is predicated upon first booting a program and then analyzing its associated registration cache and invoking the resource fork of the application to build the requisite .srg files. Thus, it is urged that Claim 24 is not obvious in view of the cited reference for at least missing claimed element (1) identified above.

As to missing claimed elements (2) and (3), and for similar reasons to those given above with respect to Claim 1, the cited reference does not teach the claimed path sequence (the path sequence *specifying an order for searching directories for locating executable code* within the data processing system), and thus is follows there is no teaching or suggestion of determining if such (missing) path sequence was modified (missing claimed element (2)), or of correcting such (missing) path sequence (missing claimed element (3)). Thus, it is further urged that Claim 24 is not obvious in view of the cited reference due to missing claimed elements (1) - (3) identified above.

With respect to Claim 26, such claim recites a method for managing environment variables in a data processing system, and is specifically directed to automatic or manual *deletion of a directory from path sequence of an environment variable*. The method includes steps of (i) determining if a directory, specified by a path sequence of any environment variable, is deleted, wherein the path sequence specifies an order for searching a plurality of different directories for locating executable code within the data processing system; and (ii) enabling at least one of the following: a) an automatic deletion of the directory from the path sequence of the environment variable, and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables. As can be seen, this claim is directed to another

technique for modifying the path sequence of an environment variable, where the path sequence specifies an order for searching a plurality of different directories for locating executable code within the data processing system. In this instance, a directory is either automatically deleted, or a user prompt for manual deletion, from such path sequence. The cited reference does not teach or otherwise suggest *deleting a directory from a path sequence* that specifies an order for searching a plurality of different directories for locating executable code within the data processing system. First, because the cited reference teaches that files are explicitly and fully specified by both directory and file name, there is no reason to have any such path sequence *as multiple directories do not have to be search to locate a file*. Second, while the cited reference teaches modification of registry keys, it does not teach or suggest the specifically claimed feature of deleting directories from any registry or registry entry. Rather, and as described with respect to blocks 345 and 355 of Figure 3, either an SReg resource is registered (page 5, paragraph [0068]), or an OTLB resource is registered (page 5, paragraph [0070]). These are the only two registry modification operations that occur when updating a registry (as taught by Hussey), and neither operation teaches or suggests any type of directory deletion from a path sequence, as expressly recited in Claim 26. The Examiner appears to acknowledge this missing claimed step, but states that such step would have been obvious "because the old directories in the environment variables is being replaced/changed to the new names and/or new locations as appropriate when the automatically updating step is performed". Applicants urge that while an explicit path reference to a given file (directory plus file name) may be changed by Hussey, such change does not teach or otherwise suggest deleting a directory from a *directory search list* (the claimed 'path sequence'), but instead teaches changing the direct reference of where the file actually resides. Thus, it is urged that Claim 26 is not obvious in view of the cited reference, as there is at least one missing claimed feature not taught or suggested by the cited reference.

Applicants traverse the rejection of Claims 27 and 30 for similar reasons to those given above with respect to Claim 24.

Applicants traverse the rejection of Claims 29 and 32 for similar reasons to those given above with respect to Claim 26.

Therefore, the rejection of Claims 1, 10, 17, 24, 26, 27, 29, 30 and 32 under 35 U.S.C. § 103 has been overcome.

B. The Examiner rejected Claims 6, 7, 9, 15, 16, 22, 23, 25, 28 and 31 under 35 U.S.C. § 103 as being unpatentable over Hussey (US Pat. Application Publication 2005/0086641), in view of Hove et al. (US 6,564,369). This rejection is respectfully traversed.

With respect to Claims 6 and Claim 7, Applicants initially traverse the rejection of such claims for similar reasons to those given above with respect to Claim 1 (of which Claims 6 and 7 depend upon), and urge that none of the cited references teach or suggest altering of a path sequence of an environment variable, where the path sequence that is altered *specifies an order for searching directories* to locate executable code; or that such altering of a path sequence *is responsive to* determining that duplicate files exist.

Further with respect to Claim 6 (and dependent Claim 7), such claim recites "wherein the step of altering the path sequence of the environment variable comprises removing references to all but one of duplicate files in the path sequence of the environment variable". As can be seen, this claim is a further refinement to the claimed *path sequence alteration* recited in Claim 1, the path sequence being altered being a path sequence that specifies an order for searching a plurality of different directories for locating executable code within the data processing system. In rejecting Claim 6, the Examiner acknowledges that the cited Hussey reference does not teach this claimed step, but states that Hove teaches a system that detects duplicate files and allows a user to move or remove duplicate files. Applicants respectfully point out that Claim 6 is not directed to a user moving or removing duplicate files. Rather, Claim 6 is directed to removing *references to duplicate files in a system variable*. An allegation of removing duplicate files does not establish a teaching or suggestion of removing references to files in a system variable, the system variable being the environment variable. Thus, the Examiner has failed to establish a prima facie showing of obviousness with respect to Claim 6[2], and thus the burden has not shifted to Applicants to rebut such obviousness assertion[3].

With respect to Claim 9, such claim recites a method for *correcting modifications that have been made to an environment variable* during installation of software in a data processing

[2] To establish prima facie obviousness of a claimed invention, all of the claim limitations must be taught or suggested by the prior art. MPEP 2143.03. *See also, In re Royka*, 490 F.2d 580 (C.C.P.A. 1974).

[3] In rejecting claims under 35 U.S.C. Section 103, the examiner bears the initial burden of presenting a prima facie case of obviousness. *In re Oetiker*, 977 F.2d 1443, 1445, 24 USPQ2d 1443, 1444 (Fed. Cir. 1992). Only if that burden is met, does the burden of coming forward with evidence or argument shift to the applicant. *Id.*

system. The method includes steps of (i) installing the software on the data processing system for subsequent execution by the data processing system, (ii) detecting that an environment variable has been modified during the installing step, (iii) responsive to the detecting step, determining if duplicate files exist in a path sequence of the modified environment variable, (iv) responsive to a determination that duplicate files exist in a path sequence of the environment variable, prompting a user to select a correct one of the duplicate files, and (v) removing all incorrect ones of the duplicate files that exist in the path sequence of the environment variable. Applicants have amended Claim 9 to further define the path sequence, and traverse the rejection of Claim 9 for similar reasons to those given above regarding Claim 1 and the missing path sequence feature.

With respect to Claims 15, 16, 22 and 23, Applicants traverse for similar reasons to those given above with respect to Claims 6 and 7.

With respect to Claim 25, such claim recites a method for managing environment variables in a data processing system, with an environment variable manager that is automatically invoked to provide a display for user correction of a path sequence based upon a determination of whether duplicate files exist in the directories specified by the path sequence of the environment variable. The method includes steps of (i) automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching a plurality of different directories for locating executable code within the data processing system; (ii) determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and (iii) enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction. As can be seen, such claim recites that *an environment variable manager is automatically invoked whenever a path sequence for a presently active environment variable is modified* in the data processing system (where the path sequence specifies an order for searching a plurality of different directories for locating executable code within the data processing system). For similar reasons to those given above with respect to Claim 24, the cited Hussey reference does not teach or otherwise suggest such automatic invocation upon occurrence of a path sequence being modified. Nor does the cited Hove reference overcome such teaching deficiency, as such reference does not teach or otherwise

suggest the claimed path sequence, and thus it necessarily follows that the cited Hove reference does not teach or otherwise suggest invoking an action when such (missing) path sequence is modified.

Still further with respect to Claim 25, none of the cited references teach or suggest the claimed step of "determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable". The Examiner acknowledges this missing claimed step, but states that since Hussey's system can determine that an operating system has been installed, it would have been obvious that "there would be duplicate files exist in the other directories which have been renamed by the user". Applicants urge that even if such assertion were true – that duplicate files exist in the system - it still does not establish a teaching or suggestion of *determining the existence of* such file duplication, but merely that it may exist in some indeterminate fashion. This distinction becomes even more apparent when viewed in combination with the next missing claimed step - enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction. As can be seen, the 'enabling a display' step is with respect to the *environment variables* (such environment variables having been determined to have duplicate files in the directories specified by the path sequence). In rejecting this aspect of Claim 25, the Examiner cites Hove as teaching a conflict system that detects duplicate files and allows a user to move or delete such duplicate files. Applicants respectfully submit that Claim 25 is different, in that *environment variables* are enabled to be displayed – and *not the duplicate files themselves*, as alleged to be taught by Hove. Thus, a prima facie case of obviousness has not been established by the Examiner in rejecting Claim 25, and accordingly the burden has not shifted to Applicants to rebut such obviousness assertion.

Applicants traverse the rejection of Claims 28 and 31 for similar reasons to those given above with respect to Claim 25.

Therefore, the rejection of Claims 6, 7, 9, 15, 16, 22, 23, 25, 28 and 31 under 35 U.S.C. § 103 has been overcome.

III. Conclusion

It is respectfully urged that the subject application is patentable over the cited references and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: September 29, 2005

Respectfully submitted,

Duke W. Yee
Reg. No. 34,285
Wayne P. Bailey
Reg. No. 34,289
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorneys for Applicants